DW_lp_multifunc

Low Power Fixed-Point Multi-Function Unit

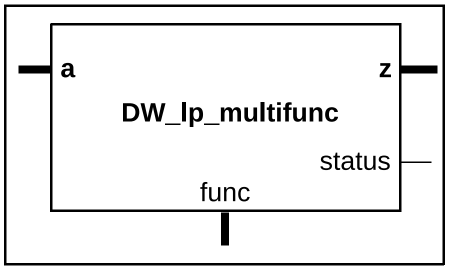
Version, STAR, and myDesignWare Subscriptions: IP Directory

Features and Benefits

Revision History

- Parameterized precision of operands and number of functions to be implemented.
- It implements any combination of the following functions:

$$\frac{1}{x}, \frac{1}{\sqrt{x}}, \sqrt{x}, \sin(\pi x), \cos(\pi x), \log_2(x), 2^x$$



- One function is computed at a time
- All functions produce monotonic results (depends on input range--see Table 1-9 on page 5)
- Shared polynomial approximation unit provides a solution with reduced area.

Note You must use VCS to simulate the DW_lp_multifunc component. Non-VCS simulators are not supported.

Description

DW_lp_multifunc is a fixed point multi-function unit that implements any combination of seven functions: reciprocal, square root, reciprocal square root, sine, cosine, base-2 logarithm and base-2 exponential. The particular set of functions to be implemented is determined with the *func_select* parameter as a one-hot value. The *op_width* parameter determines the bit-width of input and output values. At any given time the unit computes one function in the set defined by input `func`. An incorrect input value at port `func` is signaled by output `status`.

Table 1-1 Pin Descriptions

Pin Name	Width	Direction	Function
a	(*op_width* + 1) bits	Input	Input data
func	16 bit	Input	Function selection
z	(*op_width* + 2) bits	Output	Output data
status	1 bit	Output	Flag of invalid operation

Table 1-2 Parameter Description

Parameter	Values	Description
op_width	3 to 24 Default: 24	Word length of input and output
func_select	1 to 127 Default: 127	Determines the functions to be implemented among the supported functions (one bit for each function)

Table 1-3 Synthesis Implementations

Implementation Name	Implementation	License Feature Required
rtl	Low Power Synthesis model	■ DesignWare (P-2019.03 and later) ■ DesignWare-LP[a] (before P-2019.03)

a. For versions before P-2019.03, you must enable minPower as follows:
```
set_synthetic_library {dw_foundation.sldb dw_minpower.sldb}
```

Table 1-4 Simulation Models

Model	Function
dw/sim_ver/DW_lp_multifunc.v	Verilog simulation model[a] source code You must use VCS to simulate the DW_lp_multifunc component. Non-VCS simulators are not supported.

a. To use this simulation model, the '+v2k' options needs to be used on the VCS command line.

Parameter *func_select* and Input func

The *func_select* is a parameter used for the selection of the set of functions to be implemented. The valid range of the *func_select* parameter and the `func` input signal is from 1 to 127.

Table 1-5 func_select Parameter Implementation (design) and func Function Selection (use)

func_select Bit	Function Description	func 16-bit select value
2^0	reciprocal, 1/a	0000_0000_0000_0001
2^1	square root of a	0000_0000_0000_0010
2^2	reciprocal square root of a	0000_0000_0000_0100
2^3	sine, sin(πa)	0000_0000_0000_1000
2^4	cosine, cos(πa)	0000_0000_0001_0000
2^5	base-2 logarithm, $\log_2 a$	0000_0000_0010_0000
2^6	base-2 exponential, 2^a	0000_0000_0100_0000

Table 1-5 func_select Parameter Implementation (design) and func Function Selection (use) (Continued)

func_select Bit	Function Description	func 16-bit select value
	(2^7 - 2^{15}) reserved for future support	reserved for future support

For example, if only a reciprocal function is required, *func_select* needs to be 1 (16‘h0001).

If reciprocal, reciprocal square root and base-2 logarithm functions are needed, *func_select* should be 37 (16‘h0025).

If all seven functions are implemented, *func_select* is 127 (16‘h007f).

During operation, the input port `func` specifies the single function that is to be computed at that time. The `func` port is a 16-bit input port and it receives a one-hot encoded value. The valid one-hot codes for `func` are defined in Table 1-5 on page 2. If `func` receives a code for a function was not implemented by the *func_select* parameter, the `status` flag is set. If `func` does not receive a valid one-hot encoded value, the `status` flag is set. However, if only one function was implemented (*func_select* took value of either 1, 2, 4, 8, 16, 32 or 64), DW_lp_multifunc becomes independent of the input `func`, and the `status` flag is not set.

Input and Output Ranges of Functions

Valid input ranges of the seven functions are described in Table 1-6. If the input value is out of range, it turns the `status` flag on indicating an incorrect output value.

Table 1-6 Input and Output Ranges

func_select Bit	Function Description	Input Range	Output Range
2^0	reciprocal, 1/a	$1 \leq a < 2$	$0.5 < z \leq 1$
2^1	square root of a	$0.25 \leq a < 1$	$0.5 \leq z < 1$
2^2	reciprocal square root of a	$0.25 \leq a < 1$	$1 < z \leq 2$
2^3	sine, sin(πa)	$0 \leq a < 2$	$-1 \leq z \leq 1$
2^4	cosine, cos(πa)	$0 \leq a < 2$	$-1 \leq z \leq 1$
2^5	base-2 logarithm, $\log_2 a$	$1 \leq a < 2$	$0 \leq z < 1$
2^6	base-2 exponential, 2^a	$0 \leq a < 1$	$1 \leq z < 2$

Input and Output Format of Functions

Because valid input and output ranges are different among the seven functions, the number formats for the input and output ports vary. Therefore, the input format is one bit wider than *op_width* and the output format is two bits wider. The input format consists of one-bit integer part and *op_width*-bit fractional part. The output format has a two-bit integer part and *op_width*-bit fractional part. For example, Table 1-7 shows the input and output formats with *op_width* = 24. For sine and cosine functions, the output value is represented in two's complement to express signed numbers.

Table 1-7 Input and Output Ranges

func_select	Function Description	Input Format	Output Format
2^0	reciprocal, 1/a	$1.a_{23}a_{22} \ldots a_1 0$	$0\ z_{24}.z_{23}z_{22} \ldots z_1z_0$
2^1	square root of a	$0.a_{23}a_{22} \ldots a_1a_0$	$00.1\ z_{22} \ldots z_1z_0$
2^2	reciprocal square root of a	$0.a_{23}a_{22} \ldots a_1a_0$	$z_{25}z_{24}.z_{23}z_{22} \ldots z_1 0$
2^3	sine, $\sin(\pi a)$	$a_{24}.a_{23}a_{22} \ldots a_1 0$	$z_{25}z_{24}.z_{23}z_{22} \ldots z_1 0$
2^4	cosine, $\cos(\pi a)$	$a_{24}.a_{23}a_{22} \ldots a_1 0$	$z_{25}z_{24}.z_{23}z_{22} \ldots z_1 0$
2^5	base-2 logarithm, $\log_2 a$	$1.a_{23}a_{22} \ldots a_1 0$	$00.z_{23}z_{22} \ldots z_1z_0$
2^6	base-2 exponential, 2^a	$0.a_{23}a_{22} \ldots a_1a_0$	$01.z_{23}z_{22} \ldots z_1 0$

Calculation Examples

Table 1-8 shows the examples of the calculation results with *op_width* = 8 and *func_select* = 127. Numbers for input and output in Table 1-8 are binary numbers.

Table 1-8 Calculation Examples (*op_width* = 8, *func_select* = 127)

func	Expression	Input	Output
16'h0001	1/1.0111111 = 0.10101011	1_0111_1110	00_1010_1011
16'h0002	SQRT(0.11011101) = 0.11101101	0_1101_1101	00_1110_1101
16'h0004	1/SQRT(0.11011101) = 1.0100011	0_1001_1101	01_0100_0110
16'h0008	$\sin(1.1101011\pi) = -0.011111$	1_1101_0110	11_1000_0010
16'h0010	$\cos(1.1101011\pi) = 0.110111$	1_1101_0110	00_1101_1110
16'h0020	$\log_2(1.1000101) = 0.10011111$	1_1000_1010	00_1001_1111
16'h0040	$2^{0.11100111} = 1.1101111$	1_1100_1110	01_1101_1110

Error Range and Monotonicity

The calculated output value of all functions guarantees 1-ulp error range. Each function has a different ulp position depending on the valid input range. The reciprocal, sine, cosine and base-2 logarithm functions have the input range larger than 1, and the ulp position has weight $2^{-(op\_width-1)}$. The ulp position of square root, inverse square root and base-2 exponential has weight 2^{-op_width}.

The DW_lp_multifunc component produces monotonic results. The input ranges under which monotonic results are produced are shown in Table 1-9.

Table 1-9 Input Range for Monotonic Results

	1/x	sqrt(x)	1/sqrt(x)	sin(x)	cos(x)	log2(x)	exp2(x)
op_width Input Range	All values of *op_width*	All values of *op_width*	All values of *op_width*	*op_width* <= 17	*op_width* <= 17	All values of *op_width*	All values of *op_width*

Verilog Description Example

If all functions except sine and cosine functions are implemented with *func_select* = 103 (7‘b1100111) and the output needs to produce the result of the “square root” operation, the Verilog expressions needs to be described as follows.

```
DW_lp_multifunc #(8, 103) U1 (
    .a(9'b011011101),
    .func(16'h0002),
    .z(z),
    .status(status)
);
```

In this case, `z` becomes 10'b0011101101, as shown in Table 1-8 on page 4, and `status` becomes 0. However, if the above module tries to calculate the sine function with `func` = 16'h0008, it signals the `status` flag, meaning that it is an invalid operation because DW_lp_multifunc does not have the netlist for the sine function with *func_select* = 103 (7'b1100111).

Single Function Implementation from DW_lp_multifunc

By selecting the func_select parameter to choose a single function, DW_lp_multifunc generates the netlist for the single function that has the similar functionality with single-function DW components, for example, DW_sqrt, DW_invsqrt, DW_sincos, DW_log2 and DW_exp2. The difference between the DW_lp_multifunc and these single-function components is the value of 1 ulp, due to the different implementation of polynomial approximation, but all evaluated results keep the error bound less than 1 ulp. Performance and area of the synthesis results are different depending on the synthesis constraints, library cells, and synthesis environments, for example. Therefore, by comparing performance and area between the single-function implementation and the DW_lp_multifunc in single-function mode, the DW_lp_multifunc provides more choices for the better synthesis results.

The following examples generate single functions.

Fixed-point Reciprocal (U1)

```
DW_lp_multifunc #(op_width, 1) U1 (
    .a(a),
    .func(16'h0001),
    .z(z),
    .status(status)
);
```

Fixed-point square root (U2)

Results will be same as DW_sqrt except 1 ulp value.

```
DW_lp_multifunc #(op_width, 2) U2 (
    .a(a),
    .func(16'h0002),
    .z(z),
    .status(status)
);
```

Fixed-point reciprocal square root (U3)

Results will be same as DW_invsqrt except 1 ulp value.

```
DW_lp_multifunc #(op_width, 4) U3 (
    .a(a),
    .func(16'h0004),
    .z(z),
    .status(status)
);
```

Fixed-point sine and cosine (U4)

Results will be same as DW_sincos except 1 ulp value. Input `func` must be one of two values: 16'h0008 for sine and 16'h0010 for cosine evaluation.

```
DW_lp_multifunc #(op_width, 24) U4 (
    .a(a),
    .func(func),
    .z(z),
    .status(status)
);
```

Fixed-point base-2 logarithm (U5)

Results will be same as DW_log2 except 1 ulp value.

```
DW_lp_multifunc #(op_width, 32) U5 (
    .a(a),
    .func(16'h0020),
    .z(z),
    .status(status)
);
```

Fixed-point base-2 exponential (U6)

Results will be same as DW_exp2 except 1 ulp value.

```
DW_lp_multifunc #(op_width, 64) U6 (
    .a(a),
    .func(16'h0040),
    .z(z),
    .status(status)
);
```

Related Topics

- Math – Arithmetic Overview
- DesignWare Building Blocks User Guide

HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DWpackages.all;
use DWARE.DW_Foundation_comp_arith.all;

entity DW_lp_multifunc_inst is
      generic (
        inst_op_width : POSITIVE := 24;
        inst_func_select : POSITIVE := 127
        );
      port (
        inst_a : in std_logic_vector(inst_op_width downto 0);
        inst_func : in std_logic_vector(15 downto 0);
        z_inst : out std_logic_vector(inst_op_width+1 downto 0);
        status_inst : out std_logic
        );
    end DW_lp_multifunc_inst;


architecture inst of DW_lp_multifunc_inst is

begin

    -- Instance of DW_lp_multifunc
    U1 : DW_lp_multifunc
    generic map (
          op_width => inst_op_width,
          func_select => inst_func_select
          )
    port map (
          a => inst_a,
          func => inst_func,
          z => z_inst,
          status => status_inst
          );


end inst;
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_lp_multifunc_inst( inst_a, inst_func, z_inst, status_inst );

parameter inst_op_width = 24;
parameter inst_func_select = 127;


input [inst_op_width : 0] inst_a;
input [15 : 0] inst_func;
output [inst_op_width+1 : 0] z_inst;
output status_inst;

    // Instance of DW_lp_multifunc
    DW_lp_multifunc #(inst_op_width, inst_func_select) U1 (
                .a(inst_a),
                .func(inst_func),
                .z(z_inst),
                .status(status_inst) );

endmodule
```

Revision History

For notes about this release, see the *DesignWare Building Block IP Release Notes*.

For lists of both known and fixed issues for this component, refer to the STAR report.

For a version of this datasheet with visible change bars, click here.

Date	Release	Updates
January 2020	DWBB_201912.1	■ Corrected port names to be lowercase in Table 1-1 on page 1, examples, and format presentations
March 2019	DWBB_201903.0	■ Clarified some information about minPower in Table 1-3 on page 2
July 2018	DWBB_201806.1	■ For STAR 9001366625, added a note about simulator support on page 1 and in Table 1-4 on page 2 ■ Added this Revision History table and the document links on this page